Technische Universität Berlin, Berlin, 28.07.2016

Name: ..... **Matr.-Nr.:** .....

Klausur: Berechenbarkeit und Komplexität

(Niedermeier/Chen/Froese/Sorge, Sommersemester 2016)

Einlesezeit:	15 Minuten
Bearbeitungszeit:	60 Minuten
Max. Punktezahl:	50 Punkte

1	**2**	**3**	**4**	**5**	**Σ**
(12)	(10)	(8)	(12)	(8)	(50)

Allgemeine Hinweise:

- Es sind keinerlei Hilfsmittel erlaubt.
- Benutzen Sie einen dokumentenechten Stift in der Farbe schwarz oder blau. Insbesondere also keinen Bleistift, sondern einen Kugelschreiber oder einen nicht löschbaren Füller.
- Beschriften Sie jedes Blatt mit ihren Vor- und Nachnamen und ihrer Matrikelnummer.
- **Falls es in der Aufgabenstellung nicht explizit ausgeschlossen wird, so sind alle Antworten zu begründen! Antworten ohne Begründung erhalten 0 Punkte.**

Viel Erfolg!

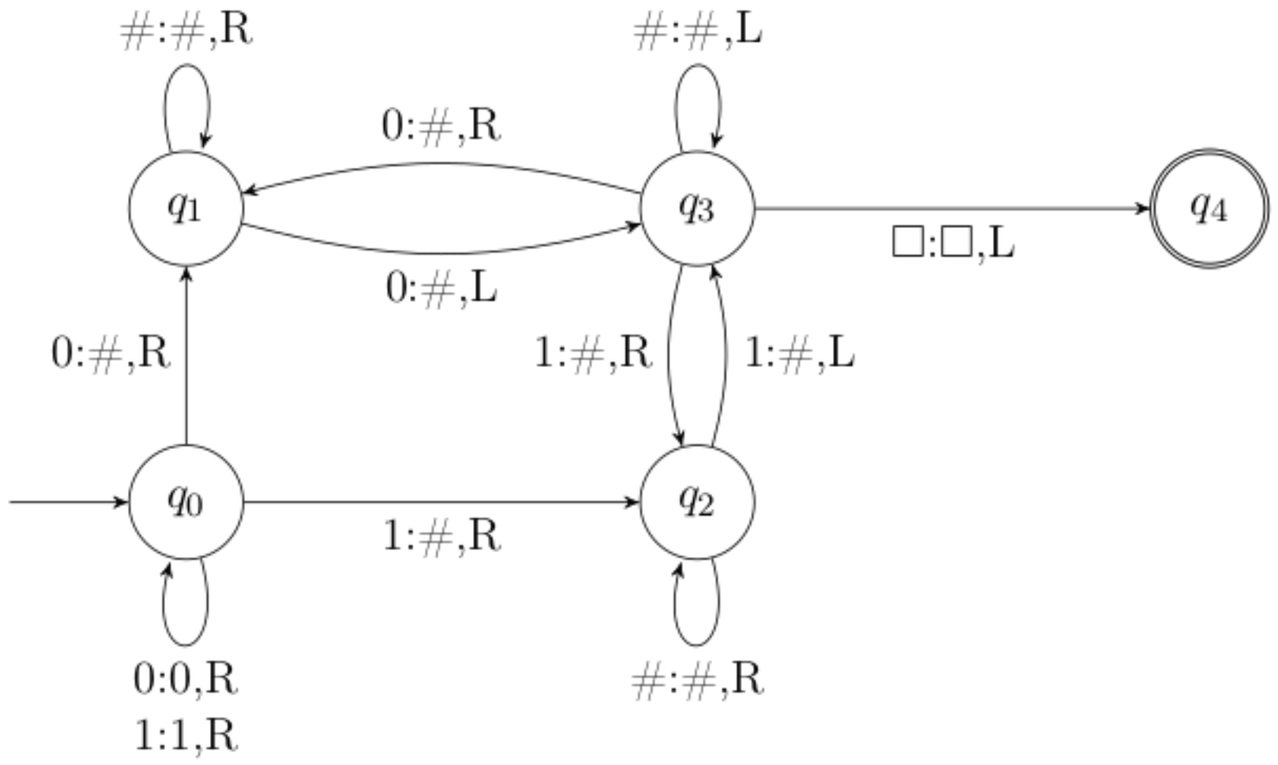
Name: ..... **Matr.-Nr.: .....**

Aufgabe 1: **Turing-Maschinen** (6 + 6 Punkte)

Betrachten Sie die nichtdeterministische Turing-Maschine

$$M = (\{q_0, q_1, q_2, q_3, q_4\}, \{0, 1\}, \{0, 1, \#, \square\}, \delta, q_0, \square, \{q_4\}),$$

wobei δ wie folgt definiert ist:



Hinweis: Beispielsweise bedeutet der Pfeil mit Beschriftung „0:#,L“ von q_1 zu q_3, dass M im Zustand q_1 beim Lesen von 0 in den Zustand q_3 übergeht, die 0 durch # ersetzt, und ihren Leseschreibkopf nach links bewegt.

a) Finden Sie ein Wort der Länge drei, das nicht von der Turing-Maschine M akzeptiert wird. Beweisen Sie, dass die Turing-Maschine M ihr Wort nicht akzeptiert.

b) Vervollständigen Sie die Konfigurationsfolge

$$q_00110 \vdash 0q_0110 \vdash \ldots$$

um zu zeigen, dass die Turing-Maschine M das Wort 0110 akzeptiert (ohne weitere Begründungen).

————Lösungsskizze————

a) M akzeptiert das Wort 101 nicht. Wir zeigen dies, indem wir alle möglichen Konfigurationsfolgen von M auf der Eingabe 101 untersuchen.

- $q_0101 \vdash \#q_201$
- $q_0101 \vdash 1q_001 \vdash 1\#q_11$
- $q_0101 \vdash 1q_001 \vdash 10q_01 \vdash 10\#q_2\square$
- $q_0101 \vdash 1q_001 \vdash 10q_01 \vdash 101q_0\square$

Für die jeweils letzten Konfigurationen ist kein weiterer Übergang definiert. In keiner der erreichbaren Konfigurationen ist M im Endzustand q_4. Daher akzeptiert M das Wort 101 nicht.

b) $q_00110 \vdash 0q_0110 \vdash 0\#q_210 \vdash 0q_3\#\#0 \vdash q_30\#\#0 \vdash \#q_1\#\#0 \vdash \#\#q_1\#0 \vdash \#\#\#q_10 \vdash \#\#q_3\#\# \vdash \#q_3\#\#\# \vdash q_3\#\#\#\# \vdash q_3\square\#\#\#\# \vdash q_4\square\square\#\#\#\#$

Name: ..... **Matr.-Nr.: .....**

Aufgabe 2: **Reduzierbarkeit und (Un-)Entscheidbarkeit** (4 + 6 Punkte)

Gegeben seien zwei endliche Alphabete Σ und Γ. Wir definieren, dass eine Sprache $L_1 \subseteq \Sigma^*$ **doppel-reduzierbar** auf eine andere Sprache $L_2 \subseteq \Gamma^*$ ist, in Zeichen $L_1 \leq_d L_2$, falls zwei totale, berechenbare Funktionen f_1 und $f_2 \colon \Sigma^* \to \Gamma^*$ existieren, so dass für alle Wörter $x \in \Sigma^*$ gilt:

$$\big((x \in L_1) \Leftrightarrow (f_1(x) \in L_2)\big) \textbf{ und } \big((x \in L_1) \Leftrightarrow (f_2(x) \notin L_2)\big).$$

Begründen oder widerlegen Sie die Korrektheit folgender Aussagen.

a) Falls L_1 doppel-reduzierbar auf L_2 ist und zusätzlich L_2 entscheidbar ist, dann ist L_1 entscheidbar.

b) Sei $H \subseteq \Sigma^*$ das allgemeine Halteproblem und sei $L \subseteq \Gamma^*$ eine beliebige Sprache. Falls H reduzierbar auf L ist, dann ist H auch doppel-reduzierbar auf L. In Zeichen:

$$(H \leq L) \Rightarrow (H \leq_d L).$$

Hinweis: Das allgemeine Halteproblem H ist semi-entscheidbar und $\Sigma^* \setminus H$ ist unentscheidbar.

————————Lösungsskizze————————

a) Die Aussage ist korrekt. Begründung: Sei L_2 entscheidbar und gelte $L_1 \leq_d L_2$ vermöge f_1, f_2. Dann ist insbesondere f_1 eine Reduktion von L_1 auf L_2, also gilt $L_1 \leq L_2$. Daraus folgt mit der Entscheidbarkeit von L_2 nach einem Satz aus der Vorlesung die Entscheidbarkeit von L_1.

b) Die Aussage ist falsch. Gegenbeispiel: Wähle $L \coloneqq H$. Offensichtlich gilt $H \leq H$. Angenommen, $H \leq_d H$ vermöge f_1, f_2. Das heißt insbesondere, dass für alle $x \in \Sigma^*$ gilt, dass

$$x \in H \Leftrightarrow f_2(x) \notin H. \tag{1}$$

Schreibe kurz $\overline{H}$ für $\Sigma^* \setminus H$. Es gilt für alle $x \in \Sigma^*$, dass

$$x \in \overline{H} \Leftrightarrow x \notin H \overset{(1)}{\Leftrightarrow} f_2(x) \in H$$

und damit $\overline{H} \leq H$ vermöge f_2. Nach einem Satz aus der Vorlesung ist dann aber $\overline{H}$ semi-entscheidbar da H semi-entscheidbar ist. Das ist ein Widerspruch, denn $\overline{H}$ ist bekanntermaßen nicht semi-entscheidbar.

Name: ..... **Matr.-Nr.: .....**

Aufgabe 3: **Satz von Rice** (4 + 4 Punkte)

Sei Σ ein endliches Alphabet und sei M_w die von $w \in \Sigma^*$ kodierte Turing-Maschine. Der aus der Vorlesung bekannte Satz von Rice lautet wie folgt:

> Sei $\mathcal{R}$ die Menge aller Turing-berechenbaren Funktionen. Sei außerdem $\mathcal{S} \subseteq \mathcal{R}$ eine nicht-triviale Teilmenge von $\mathcal{R}$, das heißt, $\mathcal{S} \neq \emptyset$ und $\mathcal{S} \neq \mathcal{R}$.
> Dann ist die Sprache $\mathcal{C}(\mathcal{S}) := \{w \in \Sigma^* \mid$ die von M_w berechnete Funktion liegt in $\mathcal{S}\}$ unentscheidbar.

Zeigen Sie für die folgenden beiden Sprachen entweder deren Unentscheidbarkeit mit Hilfe des Satzes von Rice oder beschreiben Sie die Vorgehensweise einer Turing-Maschine, die die Sprache entscheidet.

a) $L_1 \coloneqq \{w \in \Sigma^* \mid$ die von M_w berechnete Funktion ist *keine* konstante Funktion$\}$

b) $L_2 \coloneqq \{w \in \Sigma^* \mid M_w$ akzeptiert w in höchstens $|w|^2$ Schritten$\}$

————————Lösungsskizze————————

a) L_1 ist unentscheidbar. Wir beweisen das durch Anwendung des Satzes von Rice: Definiere $\mathcal{S}$ als die Menge aller nicht konstanten, Turing-berechenbaren Funktionen, das heißt

$$\mathcal{S} \coloneqq \{f \in \mathcal{R} \mid \exists x, y \in \Sigma^*. f(x) \neq f(y)\}.$$

Dann ist $\Omega \notin \mathcal{S}$, aber $\mathrm{id} \in \mathcal{S}$. Also ist $\emptyset \neq \mathcal{S} \neq \mathcal{R}$. Demnach ist $\mathcal{S}$ nicht-trivial und nach Satz von Rice folgt, dass $\mathcal{C}(\mathcal{S})$ unentscheidbar ist. Wegen $L_1 = \mathcal{C}(\mathcal{S})$ ist damit L_1 unentscheidbar.

b) L_2 ist entscheidbar. Dazu konstruieren wir eine TM M, die beim Eingabe $w \in \Sigma^*$, die kodierte TM M_w auf die selbe Eingabe w für $|w|^2$ viele Schritte simuliert. Wenn M_w das Wort w innerhalb von $|w|^2$ Schritten akzeptiert, so lassen wir M das Wort w akzeptieren. Andernfalls lehnt M das Wort w ab. Offensichtlich entscheidet M die Sprache L_2.

Name: ..... **Matr.-Nr.:** .....

Aufgabe 4: **NP-Vollständigkeit und Polynomzeitreduktion** (2 + 5 + 5 Punkte)

Betrachten Sie die aus der Vorlesung bekannten NP-vollständigen Probleme Dominating Set und Set Cover:

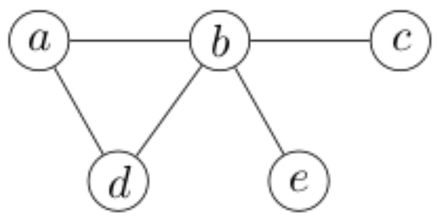
Dominating Set
Eingabe: Ein ungerichteter Graph $G = (V, E)$ und eine natürliche Zahl $k \in \mathbb{N}$.
Frage: Existiert eine Teilmenge $W \subseteq V$, sodass $|W| \leq k$ und jeder Knoten in $V \setminus W$ mindestens einen Nachbarn in W hat?

Set Cover
Eingabe: Eine Grundmenge X, eine Familie $\mathcal{F} = \{S_1, \ldots, S_m\}$, wobei jedes S_i eine Teilmenge von X ist, und eine natürliche Zahl $k \in \mathbb{N}$.
Frage: Existiert eine Teilfamilie $\mathcal{F}' \subseteq \mathcal{F}$, sodass $|\mathcal{F}'| \leq k$ und die Vereinigung der enthaltenen Mengen die Grundmenge ergibt, also $\bigcup_{S \in \mathcal{F}'} S = X$?

Für einen ungerichteten Graphen $G = (V, E)$ bezeichnen wir mit $N_G(v) \subseteq V$ die Menge der Nachbarn eines Knotens $v \in V$. Beispielsweise sind $N_G(a) = \{b, d\}$ und $N_G(b) = \{a, c, d, e\}$ im unten abgebildeten Graphen G. Betrachten Sie die polynomzeitberechenbare Funktion f, die für jeden Graphen G und jede natürliche Zahl $k \in \mathbb{N}$ wie folgt definiert ist:

$$f(\langle G = (V, E), k\rangle) = \langle \{N_G(v) \mid v \in V\}, k\rangle.$$

a) Sei $\langle G, 1\rangle$ eine Eingabeinstanz für Dominating Set, wobei G wie folgt definiert ist:



Geben Sie $f(\langle G, 1\rangle)$ an (ohne Begründung).

b) Begründen Sie anhand des Beispiels aus Teilaufgabe a), warum die Funktion f *keine* Reduktionsfunktion von Dominating Set auf Set Cover ist.

c) Korrigieren Sie die Definition von f, sodass sich eine polynomzeitberechenbare Reduktionsfunktion f' von Dominating Set auf Set Cover ergibt. Beweisen Sie anschließend die Korrektheit der Reduktion vermöge f'.

——————Lösungsskizze——————

(a) $f(\langle G, 1\rangle) = \langle \{\{b, d\}, \{a, c, d, e\}, \{b\}, \{a, b\}\}, 1\rangle$

(b) $\langle G, 1\rangle$ ist eine Ja-Instanz für Dominating Set, da eine Teilmenge $W = \{b\} \subseteq V$ existiert, sodass $|W| \leq 1$ und für alle $v \in V \setminus W = \{a, c, d, e\}$ gilt $v \in N_G(b)$.
Für $f(\langle G, 1\rangle) = \langle \mathcal{F}, k\rangle$ gilt für alle Elemente $S \in \mathcal{F}$, dass $S \subsetneq X = V = \{a, b, c, d, e\}$. Deswegen existiert *keine* Teilfamilie $\mathcal{F}' \subseteq \mathcal{F}$, sodass $|\mathcal{F}'| \leq 1$ und $\bigcup_{S \in \mathcal{F}'} S = X = V$. Also ist $f(\langle G, 1\rangle)$ eine Nein-Instanz für Set Cover.
Hieraus folgt, dass f keine Reduktionsfunktion von Dominating Set auf Set Cover ist, da wir gerade gezeigt haben, dass $\langle G, 1\rangle \in$ Dominating Set aber $f(\langle G, 1\rangle) \notin$ Set Cover.
Alternative:
Die Funktion f bildet auf *keine* gültige Eingabeinstanzen von Set Cover ab, da die Grundmenge X nicht definiert wird.

(c) Sei $f'(\langle G = (V, E), k\rangle) = \langle V, \{N_G(v) \cup \{v\} \mid v \in V\}, k\rangle$.
Wir zeigen, dass f' eine polynomielle Reduktion ist, indem wir zuerst die Korrektheit und dann die Laufzeit beweisen.

- Es ist zu zeigen, dass

$$\langle G = (V, E), k\rangle \in \textsc{Dominating Set} \Leftrightarrow f'(\langle G, k\rangle) = \langle V, \mathcal{F}, k\rangle \in \textsc{Set Cover}.$$

„⇒“: Sei $\langle G = (V, E), k\rangle$ eine Ja-Instanz für Dominating Set. Dann existiert eine Teilmenge $W \subseteq V$, sodass $|W| \leq k$ und jeder Knoten in $V \setminus W$ mindestens einen Nachbarn in W hat. Damit gilt

$$\Big(\bigcup_{v \in W} N_G(v)\Big) \cup W = V. \tag{2}$$

Setze $\mathcal{F}' := \big\{N_G(v) \cup \{v\} \mid v \in W\big\} \subseteq \mathcal{F}$. Es gilt $|\mathcal{F}'| = |W| \leq k$. Außerdem gilt $\bigcup_{S \in \mathcal{F}'} S = \bigcup_{v \in W} \big(N_G(v) \cup \{v\}\big) = \big(\bigcup_{v \in W} N_G(v)\big) \cup W \overset{(2)}{=} V$. Also ist $f(\langle G, k\rangle)$ eine Ja-Instanz für Set Cover.

„⇐“: Sei $(V, \mathcal{F}, k)$ eine Ja-Instanz für Set Cover. Dann existiert eine Teilfamilie $\mathcal{F}' \subseteq \mathcal{F}$, sodass $|\mathcal{F}'| \leq k$ und

$$\bigcup_{S \in \mathcal{F}'} S = V. \tag{3}$$

Ferner hat $\mathcal{F}'$ die Form $\mathcal{F}' = \{N_G(v_1) \cup \{v_1\}, N_G(v_2) \cup \{v_2\}, \ldots, N_G(v_{k'}) \cup \{v_{k'}\}\}$ mit $k' \leq k$.
Setze $W := \{v \mid N_G(v) \cup \{v\} \in \mathcal{F}'\} = \{v_1, v_2, \ldots, v_{k'}\} \subseteq V$. Es gilt $|W| = |\mathcal{F}'| \leq k$ und

$$\bigcup_{v \in W} N_G(v) \supseteq \Big(\bigcup_{N_G(v) \cup \{v\} \in \mathcal{F}'} N_G(v) \cup \{v\}\Big) \setminus W \overset{(3)}{=} V \setminus W.$$

Daraus folgt, dass für alle Knoten $v \in V \setminus W$, dass ein $v_i \in W$ existiert mit $v \in N_G(v_i)$. Folglich ist $\langle G, k\rangle$ eine Ja-Instanz für Dominating Set.

- Des Weiteren ist die Reduktionsfunktion f' polynomzeitberechenbar. Für eine beliebige Eingabeinstanz $\langle G = (V, E), k\rangle$ für Dominating Set werden folgende Schritte ausgeführt:
  - Für die Grundmenge X wird die gesamte Knotenmenge übernommen.
  - Für jeden Knoten in V wird die sogenannte geschlossene Nachbarschaft gebildet und als Element der Familie $\mathcal{F}$ hinzugefügt.
  - Der Parameter k wird übernommen.

  Hieraus ergibt sich eine Laufzeit in $O(|V||E|)$ und das ist polynomiell in der Eingabelänge.

Name: ..... **Matr.-Nr.:** .....

Aufgabe 5: **Vermischtes zu P, NP, und anderen Komplexitätsklassen** (2 + 2 + 4 Punkte)

Im Folgenden sei Σ ein endliches Alphabet.

a) Beweisen oder widerlegen Sie:

Wenn P = NP gilt, dann liegt jede NP-schwere Sprache $A \subseteq \Sigma^*$ in P.

b) Für eine monoton wachsende Funktion $f\colon \mathbb{N} \to \mathbb{N}$ sei DSPACE(f) die Klasse aller Sprachen $L \subseteq \Sigma^*$, für die eine deterministische Turing-Maschine M mit $T(M) = L$ existiert, sodass M für jede Eingabe $x \in \Sigma^*$ hält und höchstens $f(|x|)$ viele Zellen auf dem Band modifiziert.

Beweisen oder widerlegen Sie:

Für jede monoton wachsende Funktion $f\colon \mathbb{N} \to \mathbb{N}$ gilt DTIME(f) $\subseteq$ DSPACE(f).

c) *Zur Erinnerung*: Das Komplement $\overline{L} \subseteq \Sigma^*$ einer Sprache $L \subseteq \Sigma^*$ ist definiert als $\overline{L} := \Sigma^* \setminus L$. Die Komplexitätsklasse coNP ist definiert als coNP := $\{L \subseteq \Sigma^* \mid \overline{L} \in \text{NP}\}$. Eine Sprache $A \subseteq \Sigma^*$ heißt **coNP-vollständig**, falls $A \in$ coNP und für alle $B \in$ coNP gilt, dass $B \leq_m^p A$.

Beweisen oder widerlegen Sie:

Für jede NP-vollständige Sprache $A \subseteq \Sigma^*$ gilt, dass $\overline{A}$ coNP-vollständig ist.

————Lösungsskizze————

a) Die Aussage gilt nicht. Das allgemeine Halteproblem ist, wie auf dem 12. Aufgabenblatt gezeigt wurde, NP-schwer. Es liegt aber nicht in P, da es nicht entscheidbar ist, aber alle Sprachen in P entscheidbar sind.

b) Die Aussage gilt. Sei $L \in$ DTIME(f). Dann existiert eine deterministische Turingmaschine M, die L entscheidet und auf jeder Eingabe $x \in \Sigma^*$ maximal $f(|x|)$ Schritte ausführt. In jedem Schritt kann M maximal eine neue Bandzelle modifizieren. Daher modifiziert M auf Eingabe x maximal $f(|x|)$ Bandzellen. Also ist $L \in$ DSPACE(f).

c) Die Aussage gilt. Sei A eine NP-vollständige Sprache. Dann ist insbesondere $A \in$ NP und somit $\overline{A} \in$ coNP laut der Definition von coNP.

Es bleibt noch zu zeigen, dass $\overline{A}$ coNP-schwer ist. Sei $B \in$ coNP. Dann ist $\overline{B} \in$ NP und folglich existiert eine Polynomzeitreduktion $f\colon \Sigma^* \to \Sigma^*$ von $\overline{B}$ auf A. Das heißt, f ist in Polynomzeit berechenbar und für alle $x \in \Sigma^*$ gilt $x \in \overline{B} \iff f(x) \in A$. Folglich gilt auch $x \in B \iff f(x) \in \overline{A}$. Somit ist f auch eine polynomzeitberechenbare Reduktion von B auf $\overline{A}$. Damit ist gezeigt, dass sich jede beliebige Sprache in coNP in Polynomzeit auf $\overline{A}$ reduzieren lässt. Also ist $\overline{A}$ coNP-schwer.